Implementation and Analysis of Graph Algorithm Used for Software Plagiarism Identification
Sowgandh Krishnaa Nandamuri (sn2638@g.rit.edu)
Advisor: Dr. Carlos Rivero
Rochester Institute of Technology

Introduction
Software Plagiarism occurs at various levels varying from schools to industries.
In Schools it occurs in programming courses taken by students, while in the industries it occurs during rapid product development with the source for plagiarism being freely available open source projects.
Core-Plagiarism : Plagiarism occurring in a small portion of a very large project.
The project deals with the situation, by constructing graphs over procedures or functions of a program known as Program Dependence Graphs (PDG's).

Previous Work
Text-Based : Programs are considered as sequences of text. Main drawback is that these techniques do not take into consideration the syntax of language, susceptible to identifier renaming and alternate syntax.
Token-Based : Programs are parsed into a set of tokens based on syntax of language or some logic. Plagiarism is reported if level of similarity is greater than threshold. Unable to recognize alternate control structures.
Tree-Based : Programs are parsed into trees using lexers and parsers. The variable and literal names are ignored, subtrees of plagiarized code are searched for the in the original code.

Objective
To implement a code plagiarism detection tool by constructing PDG’s over procedures of original and plagiarized program and checking if they are subgraph isomorphic to one another.

System Architecture

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Name</th>
<th>UseCase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decl</td>
<td>Declaration</td>
<td>Eg int i, int h=k</td>
</tr>
<tr>
<td>Control</td>
<td>Control structure</td>
<td>Used in place of, if while, for loop etc</td>
</tr>
<tr>
<td>Assign</td>
<td>Assignment</td>
<td>Eg int y=10, char u=3</td>
</tr>
<tr>
<td>Inc/dec</td>
<td>Increment/decrement</td>
<td>Eg i++ or --k</td>
</tr>
<tr>
<td>Expr</td>
<td>Expression</td>
<td>General expression</td>
</tr>
</tbody>
</table>

MOSS and JPLAG
- MOSS and JPLAG are two popular tools that are being used currently to test Plagiarism. Both MOSS and JPLAG are token-based systems.

Results

Plagiarism Disguises
- Random Code Insertion
- Alternate Control Structures
- Identifier or literal Renaming
- Rearrangement of statements

Conclusion
Thus a graph based plagiarism system was implemented and tested with real world programs obtained from websites such as CodeChef and CodeForces.

Future Work
The tool can be made more effective, if it is able to differentiate user defined method calls and method calls that come along with the libraries itself. Construction for the entire program, this would include smaller graphs for functions present within the program. Such a structure would help us take user defined function calls also into account.

REFERENCES
[1] Philip S. Yu. , ChaoLiu, Chen, Jiawei Han, Detection of Software Plagiarism by Program Dependence Graph Analysis. Industrial and Government Application Paper, 2006.